NASA-TM-111509

Automated Management and Scheduling of Remote Automatic Telescopes

John Bresina
Mark Drummond
Will Edgington
RECOM Technologies, Inc.
Kieth Swanson
NASA

Artificial Intelligence Research Branch
NASA Ames Research Center
Mail Stop 269-2
Moffett Field, CA 94035-1000

NASA Ames Research Center
Artificial Intelligence Research Branch
Technical Report FIA-93-28
October, 1993

Automated Management and Scheduling of Remote Automatic Telescopes

John Bresina
Recom Technologies

Mark Drummond
Recom Technologies

Keith Swanson
NASA

Will Edgington
Recom Technologies

AI Research Branch, Mail Stop: 269-2
NASA Ames Research Center
Moffett Field, CA 94035-1000

E-mail: `xfr@ptolemy.arc.nasa.gov`

Abstract

This paper describes our project on advanced planning and scheduling for remote, fully automatic telescopes. In addition to a more advanced automatic scheduler for nightly observations, our project is also building automated tools to address the entire life-cycle of an observation request, from original receipt to the return of raw data and preliminary data reduction. Our focus is on providing software tools to help a telescope manager who represents a community of participating astronomers; however, the increased automation also improves the way in which the astronomers interact with this manager. Our goal is to make it possible for participating astronomers to submit observation requests and obtain results from a remotely located telescope, via electronic networks, without the necessity of human intervention.

This paper appears in *Astronomy for the Earth and Moon*, the proceedings of the 103rd Annual Meeting of the Astronomical Society of the Pacific, 1993, edited by Diane Pyper Smith.

Automated Management and Scheduling

Recom Technologies | *Recom Technologies*

Keith Swanson
NASA

Will Edgington
Recom Technologies

AI Research Branch, Mail Stop: 269-2
NASA Ames Research Center
Moffett Field, CA 94035-1000
E-mail: `xfr@ptolemy.arc.nasa.gov`

Abstract

This paper describes our project on advanced planning and scheduling for remote, fully automatic telescopes. In addition to a more advanced automatic scheduler for nightly observations, our project is also building automated tools to address the entire life-cycle of an observation request, from original receipt to the return of raw data and preliminary data reduction. Our focus is on providing software tools to help a telescope manager who represents a community of participating astronomers; however, the increased automation also improves the way in which the astronomers interact with this manager. Our goal is to make it possible for participating astronomers to submit observation requests and obtain results from a remotely located telescope, via electronic networks, without the necessity of human intervention.

1 Introduction

Telescopes have always been a scarce resource, especially high quality telescopes located at prime observatory sites. Astronomers have had to make do with limited access, usually allocated in contiguous "blocks" of time. Further, astronomers have been expected to be physically present at an observatory in order to gather data. Limited access, block allocation, and local operation have restricted both the amount of data that can be gathered and the type of observational campaigns that can be accomplished.

For example, Fairborn Observatory and AutoScope Corporation have designed and built software and hardware systems for the management and control of modest-aperture photoelectric telescopes. These systems make it possible for a remotely located telescope to operate unattended for significant periods (months).

While the majority of existing ground-based automated telescopes are used for aperture photometry, automation support for spectroscopy and imaging has been increasing (primarily due to the efforts of R. Kent Honeycutt and Don Epand). Hall and Genet (1988) give an excellent overview of photometry, and Genet and Hayes (1989) describe automatic photoelectric telescopes in some detail.[1]

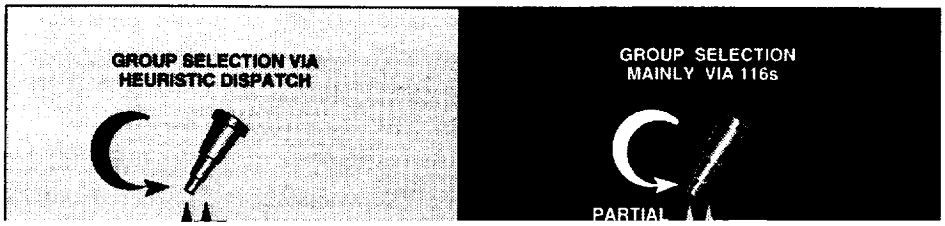
The language used to define observation requests is the Automatic Telescope Instruction Set, or ATIS (Boyd, *et al.*, 1993). In ATIS, a *group* is the primitive unit to be scheduled and executed. A group is a sequence of telescope commands and instrument commands defined by an astronomer. In the initial version, ATIS89, the only instruments accommodated were photometers, but the most recent version, ATIS93, also includes commands to utilize CCD cameras. Astronomers use the group as the primary unit of instruction to an automatic telescope in order to achieve their scientific goals. In addition to specifying the syntax and semantics for observation requests and results, the ATIS standard provides a set of *group selection rules* that are used to determine execution order of groups during the night.

While the group selection rules provide an ATIS-compatible telescope with a native observation scheduler, we thought that it should be possible to do better with more sophisticated scheduling techniques. By "better", we mean more observations per night, increased quality of the observations, and fairer allocation of telescope time to multiple participating astronomers. We were invited to be part of the International Astronomical Union ATIS93 standardization committee to assist with ATIS extensions in support of advanced scheduling. Along with other committee members, we designed a new group selection advice statement. The committee also agreed on a mechanism for communication with a telescope controller in terms of incremental ATIS93 partial input and partial output files. Together, these new features make it possible to implement a "non-native" (*i.e.*, external) scheduler that can effectively drive a telescope's controller to better serve the scientific objectives of participating astronomers.

This paper introduces our project on advanced planning and scheduling for automatic telescopes. We explain how the new features of ATIS93 allow our scheduler to communicate with an ATIS-compatible telescope controller. While the technical focus of our project is on automation for planning and scheduling, we take a broader view on the role of automation in general. In addition to an automatic observation scheduler, our project is also building automated tools to address the entire ATIS life-cycle. Our goals are to provide software tools to assist managers of multi-user telescopes and to make it possible for participating astronomers to submit observation requests and obtain results from a remotely located telescope, via electronic networks, without the necessity of human involvement.

We begin the next section with an explanation of the scope and goals of our project. Following this, we explain how our system schedules observation requests using the ATIS93 advice statement, and we show how ATIS93 partial input and output files allow our scheduler to communicate with a telescope controller. This paper concludes with a summary of our progress to date and provides a sketch of where we are going with this work.

[1] For additional background material regarding automatic telescopes and their use, see the chapter by the Genets and the chapter by Pyper Smith (both in this volume).


GROUP SELECTION VIA
HEURISTIC DISPATCH
GROUP SELECTION
MAINLY VIA 116s
PARTIAL

The groups are executed by the telescope controller (using the group selection rules) for some number of nights (weeks, perhaps months); eventually, the PA requests from the controller the results that have been collected thus far. The elapsed time varies depending on the telescope, the groups, the PA, the participating astronomers, and a variety of other factors. The collected data is returned to the PA as a "results file" specified with the ATIS language. The results include the raw data obtained from the observations, as well as, a chronological record of the groups that were executed, and relevant observing parameters to help with data reduction. The PA edits the results file and sends each astronomer the pieces corresponding to the requested observations (again typically via e-mail or floppy discs).

This current level of automation in the management and use of automatic telescopes provides an excellent starting point; we want to further improve this process by providing, through increased automation, a "simplified management structure". The term refers to an approach to the management and control of telescopes that minimizes the number of people that must come between an astronomer's scientific goals and the telescopes required to realize those goals. Our project aims to provide automation support for all aspects involved in the use and management of ATIS-compatible telescopes. Our focus is on providing software tools to help a Principal Astronomer who represents a community of participating astronomers; however, the increased automation also improves the way in which the astronomers interact with a PA. The right half of Figure 1 and the following scenario illustrate a new way of doing business with automatic telescopes that we are in the process of making possible.

From the perspective of a participating astronomer

An astronomer creates an ATIS93 observation request file and sends the file via electronic mail to the principal astronomer's computer. Let us refer to this computer as the "Associate Principal Astronomer", or APA. The mailed file is automatically received and parsed to check for syntactic errors. If the file adheres to the ATIS93 specification, then the APA e-mails a message back to the astronomer acknowledging successful receipt of the request file; otherwise, a message is e-mailed back identifying the syntactic errors in the astronomer's file. At the end of each observing night, the APA e-mails the astronomer the results of those observation requests that were serviced that night, along with the results necessary for data reduction and data quality assessment. Based on these nightly results, the astronomer can choose whether or not to modify the observation request file.

From the perspective of a principal astronomer

After the cutoff time for new observation requests, the principal astronomer checks the APA for new files that have been parsed successfully. If necessary, a new combined ATIS93 file is automatically assembled from all the requests of the telescope's user community. The PA can check how the controller will handle this new composite request file by displaying a prediction of the telescope behavior for the night based on the best schedule found by our scheduler (*i.e.*, what observations are likely to be made if the weather is ideal). If the PA is not satisfied with the prediction, then the manner in which the APA schedules the observations can be modified. Once satisfied with the predicted observation schedule, the PA's job is done for the day (and possibly done until the ATIS input file changes dramatically). The next morning, the results of the night's observations are already stored at the APA. If the PA wants to assess the quality of the night's observation schedule and results, the actual telescope behavior can be displayed. Once the PA has tuned the scheduler to consistently produce high quality schedules for the mix of ATIS input received, the APA can take care of the day-to-day management with only occasional supervision by the PA.

From the perspective of telescope operations

Just before the observation night begins, the ATIS93 input file is automatically transferred to the telescope controller along with the observation schedule. The controller executes the schedule and at the end of the observation night, transfers the ATIS93 output file back to the APA. This is the minimum amount of interaction between the controller and the APA; however, the ATIS93 specification also allows for partial input and partial output files to be transmitted during the night. The partial output files enable the telescope behavior and status to be monitored during the night – either by a person (for example, to check the status of the telescope mechanics and optics) or automatically by the APA. The partial input files enable the APA to transmit new schedules and new groups during the night when necessary. For example, the APA could dynamically reschedule due to a change in the quality of observing conditions or due to an urgent observation request received during the night.

3 Advanced Scheduling

First, we briefly describe how ATIS groups are presently scheduled by the ATIS group selection rules (for more details, see Genet & Hayes, 1989); this is also the default behavior of an ATIS93–compatible telescope controller if the APA does not send a schedule. We next contrast this method of scheduling with the method used by the APA and explain how the APA–generated schedules are used during the telescope controller's group selection process.

3.1 Dispatch Scheduling

The overall structure of every ATIS–based telescope controller is a sense-select-execute loop. First, all relevant observing parameters are sensed (current time, whether the moon is up, *etc.*). Second, based on the sensed parameters, the controller selects a group to execute next. Third, the controller executes the selected group. Under normal circumstances, this cycle repeats throughout the night.

Selection of the next group to execute occurs as follows. The controller first determines the set of "enabled" groups. A group is *enabled* if its preconditions (specified by the requesting astronomer) are satisfied with respect to the current sensed parameters. The set of enabled groups is then winnowed by the application of the ATIS group selection rules. These rules capture heuristic knowledge about which group to execute next. In scheduling parlance, this scheme is often called *heuristic dispatch*, since at any point in time, some task (here, a group) is "dispatched" for execution, and the selection of a task is determined, purely locally (without look-ahead), by the application of domain-specific heuristics.

The group selection rules reduce the set of enabled groups to a single group to be executed next. There are four ATIS group selection rules that are applied in the following sequence: *priority*, *number-of-observations-remaining*, *nearest-to-end-local-sidereal-time*, and *file-position*. If there is only one group remaining after applying any rule, then that group is selected and no further rules are applied. Since no ties can exist after the *file-position* heuristic, application of the group selection rules deterministically chooses a unique group.

How well does this type of observation scheduling perform? It is clear that the heuristic dispatch rules do provide a reasonable level of performance for some situations. However, it should be possible to improve telescope utilization through better group scheduling. With the heuristic dispatch technique, all decisions are *local* in the sense that no temporal look-ahead is performed to evaluate the ramifications (on the entire night's schedule) of executing a given group at a particular time. For example, without look-ahead, there is no way to predict that a particular group selection decision will cause the telescope to be idle for some period of time later in the night (whereas, with different choices, it would have been possible for the telescope to be fully utilized). Furthermore, with dispatch scheduling, the PA does not have an effective way of influencing, or even predicting, how scheduling decisions are made. Currently, the primary method for the PA to influence group scheduling is to alter the priorities specified in the group headers, and the impact of such [illegible]

3.2 Look-ahead Contingent Scheduling

execution involves the real world, and the real world introduces uncontrollable exogenous factors (such as winds and clouds) which conspire against error-free schedule execution.

CERES makes assumptions regarding exogenous factors; specifically, the scheduler assumes that there will be no clouds and no winds to prevent successful star acquisition.[2] Any given schedule represents an expectation that can fail. If clouds or winds make star acquisition impossible, then the schedule fails – the desired sequence of groups cannot be realized by the telescope controller.

When the current schedule fails, the controller invokes CERES to generate a new one. CERES dynamically reschedules, taking into account observations made so far. The protocol for this dynamic rescheduling process is discussed in Section 4. For now, the important point is simply that while weather can cause a failure of schedule execution, the system is robust enough to simply reschedule and try again. Of course, if the night is completely clouded over, there is not much that CERES can do, no matter how often it reschedules.

There are certain types of execution errors that can be managed so as to reduce the amount of dynamic rescheduling. Some execution errors are caused by a *duration prediction* error on the part of the scheduler. This can happen as follows. The scheduler looks at each group and calculates how long the group should take to execute. This calculation includes star acquisition times, observation integration times, and telescope slew times. Since star acquisition and telescope slew can not be predicted with certainty, the group duration is only an estimate. The start time of a group in the schedule is based on the sum of the estimated durations of all groups that precede it. Hence, the further into the future a group occurs in the schedule, the greater the uncertainty surrounding its scheduled start time.

Recall that each group can only be executed within a specific local sidereal time window, as specified by an astronomer. Given the way that uncertainty grows into the future, it is possible that CERES will produce a schedule that calls for a group to be executed at a time outside this window. Since the controller will only execute a group at a time within its window, a schedule can fail during execution due solely to these duration prediction errors.

We have developed a technique called *Just-In-Case* scheduling (JIC) to manage such execution errors. The basic idea behind JIC is to anticipate possible execution errors and generate contingent schedules to manage them just in case they do occur. While dynamic rescheduling could also recover from such execution errors, this can waste valuable telescope time during the night. JIC uses an explicit model of the uncertainty in a group's duration. This model is factored into all group start-time predictions, giving the scheduler a degree-of-confidence measure for each group's executability. Instead of a single group sequence, the schedule that JIC produces is a *tree* of contingent group sequences (for an example of this, see Figure 2). During execution, if the next scheduled group is outside its time window, the controller follows one of the contingent branches of the tree. An example schedule execution trace is presented in the next subsection.

We are currently experimentally evaluating the performance of JIC in terms of costs and benefits. The cost of JIC is increased computation during scheduling and greater memory requirements to store the contingent schedules. The benefit of JIC is less dynamic rescheduling and faster response time to execution errors. Our experiments should help us better understand the true utility of JIC with respect to the management of automatic telescopes.

[2] A cloud prevents star acquisition and observation in the obvious manner; wind can prevent automatic acquisition by causing the telescope to shake.

3.3 Schedules as Advice Statements

The purpose of the ATIS93 group selection advice statement, called the "116", is to provide a means for an external scheduler to influence the group selection process. The group selection process for ATIS93 must first check whether or not it has a currently executable schedule; that is, whether there are any currently applicable group selection advice statements. If there are applicable 116 statements, then their advice is followed; otherwise, the default ATIS group selection rules are used to select the next group to execute. Thus, the scheduling capability provided by the new group selection advice statement augments, but does not replace, the previous method of group selection.

The 116 advice statement was initially based on a representation we had previously used called "situated control rules" (Drummond, 1989). The current 116 advice statement specification was developed in collaboration with Louis Boyd, of Fairborn Observatory, and forms a more expressive schedule representation than situated control rules. In this section we discuss only the aspects of the advice statement necessary to briefly explain how schedules are encoded by CERES and executed by the controller; see the Appendix for further details on the syntax and semantics of the 116 statement.

Briefly, the syntax of the group selection advice statement is as follows (see Figure 3 for an example). Each statement consists of two lines; the first contains 116, and the second one contains the following thirteen arguments.

1 Advice Number
2 Start LST (local sidereal time)
3 End LST
4 Start UT (universal time)
5 End UT
6 Previous Group
7 Set Execution Count
8 Group Number
9 User Number
10 Group Test
11 Next If True
12 Next If False
13 Wait Flag

The first argument is an identifier for the 116 statement (that can be referred to in the Next-If-True or Next-If-False arguments of other 116 statements). The next five arguments (2–6) define applicability conditions. Specifically, arguments 2 and 3 define a time window such that the group can only be executed if the current LST is greater than the value given by argument 2 and less than the value given by argument 3. Arguments 4 and 5 similarly define a different time window; in this case, the values are given in terms of universal time (UT). If both LST and UT windows are defined for a group, then both must be satisfied in order for the group to be executable. A pair of zeros for a window indicates a "don't care" condition, in which case the window precondition is ignored.

The seventh argument of the 116 statement specifies how to update the group's execution history. The eighth and ninth uniquely identify the group to execute. The eleventh and twelfth encode *go-to*'s; *i.e.*, they specify which 116 statement to consider next, depending (respectively) on whether the current 116 is applicable (true) or not (false). The last argument specifies whether

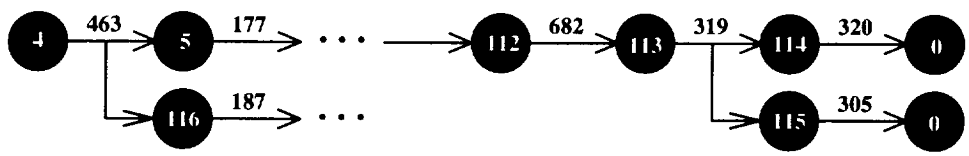


Figure 2: Tree of contingent schedules. The circles represent states labeled with the number of the advice statement that applies to the state. The arrows represent (predicted) group execution labeled by the recommended group number.

```
  *                                             *            *   *

                                                ⋮

116
  4 18.394 19.094  0.000  0.000   0  -1 463   30 1   5   0 0

116
  5 17.863 18.813  0.000  0.000   0  -1 177   30 1   6 116 0

                                                ⋮

116
112  0.370  4.730  0.000  0.000   0  -1 682   43 1 113   0 0

116
113  1.980  2.930  0.000  0.000   0  -1 319   30 1 114 115 0

116
114  2.980  3.930  0.000  0.000   0  -1 320   30 1   0   0 0

116
115  2.931  3.849  0.000  0.000   0  -1 305   30 1   0   0 0

116
116 18.814 19.139  0.000  0.000   0  -1 187   30 1 117   0 0

                                                ⋮
```

Figure 3: Fragment of contingent schedule generated by CERES. The schedule represented by these 116 statements is illustrated in the figure above; the 116 arguments in the starred columns are used in the illustration.

the controller should wait until the time is past the start LST and start UT before proceeding.

In their most basic form, 116 statements enable the encoding of *if-then* advice rules, where the antecedent specifies the applicability conditions of the advice and the consequent specifies a group to execute. Note that these applicability conditions are different than the group's preconditions specified by an astronomer – a group's preconditions specify when the group *can* be executed and the applicability conditions of a 116 statement specify when the recommended group *should* be executed (in order to yield a good schedule).

A schedule is *executable* at a point in time if there is an applicable 116 statement. If there is no such 116 statement, then the schedule is said to have "failed".

Consider the fragment of a schedule generated by CERES illustrated as a tree of contingent schedules in Figure 2; the actual 116 statements are shown in Figure 3. The following describes how the telescope controller might execute this contingent schedule. For this example execution trace, we assume that the first three advice statements (not shown in the figures) were used successfully, and that the controller follows the recommendation of advice number 4 and executes group 463.

The controller then evaluates the applicability of advice number 5, which recommends group 177. If the current time is outside group 177's LST window then advice number 5 will not be applicable. In this case, the controller tries to use the contingent schedule that starts with advice number 116, which recommends group 187 instead.

However, for this example execution, let's assume that advice number 5 was indeed applicable and that the controller was able to continue following the schedule up through advice number 113. Following the execution of group 319, the controller attempts to use advice number 114. If the advice is applicable, then group 320 is executed; otherwise, the controller tries to use advice number 115 which recommends group 305. Following execution of either group 320 or group 305, the schedule ends (indicated by a zero advice number).

4 APA–Controller Interaction

In this section, we describe our current design and implementation for interaction between the telescope controller and the APA. We also mention some possible modifications and alternatives. The objective of the interaction protocol design was to ensure that the addition of the advanced scheduling component did not degrade, in any way, the ability of the controller to gather data throughout the observing night.

As mentioned in Section 2, at the beginning of the observing night the ATIS input file and the initial schedule are automatically transferred from the APA to the controller. At the end of the observing night, the ATIS output file is automatically transferred from the controller to the APA. During the night, partial input files can be transferred to the controller and partial output files can be transferred to the APA.

Currently, the transfer of partial input and output files occurs only when the schedule fails during execution. This is implemented as follows. The new group selection process first checks whether there is an applicable 116 statement to use. If there is not, then the controller initiates a file transfer to the APA; all the stored partial output files (one for each group that has been executed) are transferred to the APA. From these files, the APA can track what part of the schedule has been executed and can ascertain the current state of the telescope and observing environment. The controller then does one round of heuristic dispatch; that is, it uses the ATIS group selection rules to determine the next group to execute.

While this group is executing, CERES reschedules; that is, it generates a new schedule starting from the predicted state that will result when the current group execution is finished. CERES predicts how long the group's execution will take and uses this to limit the amount of time spent scheduling so that a new schedule will be ready for the controller. The amount of time it takes to execute one group (*e.g.*, a differential photometric group takes on the order of 10 minutes) is plenty for CERES to develop a good schedule.

Upon completion of the group execution, the controller contacts the APA again and retrieves the new schedule (*i.e.*, the new partial input file). The 116 advice statements in this new schedule will then be used to select groups to execute until the end of the observing night or until the schedule again becomes unexecutable. Since CERES reschedules concurrently with group execution, the wasted observation time due to schedule failure is minimized.

In the unlikely event that the communication link between the telescope controller and the APA breaks down, we do not want the controller to waste time trying to get a new schedule from the APA after each group execution. Hence, we allow the PA to set a limit on the number of consecutive communication failures. Once this limit is exceeded, the controller no longer tries to transfer files to or from the APA, and the controller uses the default ATIS group selection rules to select groups for the rest of the night.

Initiating file transfers only when the schedule fails is our current base-line approach. Other options include initiating file transfers after some number of groups have been executed (*e.g.*, after every 10 groups) or after some amount of time has passed (*e.g.*, every hour); also, various combinations of these three methods could be used. The appropriate frequency of interaction between the controller and the APA depends on such factors as the type of communication link between the controller and the APA and the degree of real-time flexibility required during the night. For example, if the APA is located far from the controller and is connected via phone line and a slow modem, then the frequency of interaction should be minimized. However, if the APA is in the same warm room as the controller and is connected over a high-speed, reliable local area network, then interaction can be as frequent as desired. Between these two extremes, there are other possible configurations.

As described above, all the file transfers are initiated by the controller. This is due to current technical limitations on the controller; in the future it will be possible for either the controller or the APA to initiate file transfers. This bi-directional communication will enable the APA to receive an urgent observation request during the night, include the request in an updated schedule, and send the new groups and 116 advice statements to the controller (without having to wait for the controller to ask for a new schedule). This communication facility is crucial for the flexible use of distributed networks of automatic telescopes (see the chapter by Mason in this volume).

5 Concluding Remarks

In this section, we summarize current project status, outline some near-term and long-term goals, and conclude.

5.1 Current Status

In an early phase of our project, we developed tools that provide basic data management capabilities for browsing and editing a summarized form of raw ATIS. These initial tools have been modeled after the widely-used program CREATE that was designed and developed by George McCook of Villanova University. We are currently working on new versions of some of these tools and are coding them in C with an X11 graphics environment to ensure easy portability.[3]

The first publically available tool is a parser for ATIS89 and ATIS93; this is one component of our fully automated ATIS submission system. This parser is able to verify the syntactic correctness of ATIS files and perform partial semantic checking. Depending on the command line options and output specification, the parser will either echo the input files to the output or only print the warnings and errors encountered (in the fashion of most programming language compilers). The output is normally directed to the screen or terminal, but can be redirected to a file or multiple files (*e.g.*, one output file per input file). Any of the warnings or errors can be disabled via command line options.

The parser itself is implemented using two standard UNIX tools: **yacc** (a compiler generator) and **lex** (a lexical analyzer generator). Both tools produce C code that should be easy to compile with most any C compiler, whether UNIX, DOS, VMS, or another operating system. Presently, the code has been tested only on a Sun SPARCstation using three Free Software Foundation tools: **bison** (a slight improvement on yacc), **flex** (a faster version of lex), and **gcc** (a C compiler). The parser has been tested, and all of the parser's code, including not only the **yacc** and **lex** but also the generated C, will soon be released for public use and evaluation. Please contact **wedgingt@ptolemy.arc.nasa.gov** via e-mail if you would like to evaluate the current version of the ATIS parser.

We have also made significant progress in modifying the ATIS controller that was originally developed by AutoScope (1991). These modifications allow the telescope controller to interact with a scheduler using ATIS93 group selection advice statements and partial ATIS files as described in the preceding sections. Our upgraded version of the controller will be used by AutoScope as the starting point for their development of a fully compliant ATIS93 controller.

Using a high fidelity telescope simulator developed by AutoScope (Genet, 1992), we have tested the coordinated operation of our modified version of the AutoScope controller and CERES over numerous simulated nights of observation. Preliminary scheduling results have been quite encouraging and experiences with our scheduler–controller communication implementation have been excellent. This interaction mechanism has allowed us to perform on-line monitoring and dynamic rescheduling of simulated telescope operations using the new facilities provided by ATIS93. We have also developed preliminary graphics tools for real-time monitoring of telescope operations and are currently working on making some of these tools available as stand-alone modules.

In July of 1993, NASA awarded a Small Business Innovation Research Phase *II* contract to the AutoScope Corporation; this funding will allow our collaboration with AutoScope to continue and will result in a state-of-the-art 20 inch fully automatic telescope. Before this telescope is operational, we will test our prototype APA system on an existing AutoScope 10 inch telescope. In order to test our advanced scheduling and increased automation on the 10 inch and 20 inch telescopes, we are working with astronomers who are well-versed in the current way of managing ATIS-based telescopes.

In September of 1993, we successfully tested the interaction between the scheduler running on a Sun SPARCstation at NASA Ames and the modified controller (in simulation mode) running on a PC at the Mt. Wilson Observatory. The communication between the two computers was implemented

5.2 Future Plans

We are collaborating with Cindy Mason (an NRC Fellow at NASA Ames) on a project involving the use of CERES in a distributed global area network of automatic telescopes. In this paper, we have described the functionality of a *stand-alone* APA. This functionality must be augmented for a *network-enabled* APA (*i.e.*, in order for it to participate in a distributed network of multiple APAs). For details on this work, see Mason's chapter in this volume.

In the near term, we intend to continue the empirical evaluation and improvement of our Just-in-Case scheduling technique. In addition, we intend to continue our work on producing tools for the automatic receipt of observation requests and automatic mailing of observation results. We also intend to improve and release tools that enable the PA to view and evaluate both the predicted telescope behavior (based on a night's schedule) and the actual behavior (based on the ATIS output).

One of the long term goals of our project is to augment our advanced scheduling tools with advanced planning tools. In service of this goal, we have done some preliminary work jointly with other scientists (principally with Louis Boyd and Russell Genet) on the definition of a more expressive, higher-level language for specifying observation programs and scientific campaigns.

While ATIS provides a means for defining the structure of the individual observations within a group (*e.g.*, an observation sequence used for differential photometry), it does not provide a means for defining the structure of a set of groups that constitute a particular observation program (*e.g.*, filling out a light curve for a particular Cepheid variable). The subset of groups that define a particular observation program are not explicitly distinguished from the groups that are part of other observation programs. Furthermore, the ATIS input contains no explicit description of the overall scientific goals of the observation programs, nor of the constraints and dependencies that exist between the observations.

For example, an astronomer may want to observe a particular variable star every three to five days; this constraint on the gap between observations influences how the astronomer specifies the set of ATIS groups. However, whether or not data will actually be collected every three to five days is dependent on how the telescope is loaded and how the groups are scheduled; yet neither the loader (currently the PA) nor the scheduler are given an explicit statement of the desired gap between observations.

Automating more of the tasks of the PA and the participating astronomers will require that such factors be expressed explicitly as part of the observation program specification. We have begun a research project on applying advanced planning techniques to the automation of two such tasks: telescope loading (in support of the PA) and observation planning (in support of astronomers).

Telescope loading involves determining the subset of groups to assign for a given night (to then be scheduled by CERES during a more restricted time interval); it could also involve adding groups for quality control. Whereas the temporal scope that CERES reasons over is one night, telescope loading requires reasoning over multiple nights or, perhaps, an entire observing season. The following scenario sketches how automated observation planning and telescope loading might be used.

Astronomers submit a high level description of their long-term scientific campaigns, stating their goals, observing constraints, quality constraints, tolerances, *etc.* The automated observation planning system translates each campaign into a set of ATIS groups that satisfies the campaign's goals and constraints. The automated loader assigns the groups to particular nights of the observing season. Then as the season progresses, the results from each night's observations are used for replanning and reloading. Replanning modifies a campaign's set of groups to take into account what data has been successfully collected. For example, the relative amount of collected data may vary for different phase intervals of a star's light curve and replanning could dynamically shift the focus to those phase intervals with less data. Reloading modifies the future nights' assignments to account for groups that did not get executed (due to, for example, weather or scheduling constraints).

5.3 Conclusion

The overall goal of our project is to provide automation support for the management and use of remote, automatic telescopes. So far, we have focused on building the core of an Associate Principal Astronomer, or APA. This core consists of an automatic group scheduler and schedule execution mechanism. While this core provides important functionality, there are many aspects of the PA's job that it does not address. In collaboration with astronomers, we are currently expanding the set of functions offered by the APA to include automatic handling of ATIS request files, preliminary "quick look" data reduction, and quality control measures. Experience gained with simulation tests has been encouraging, and we are now ready to test the system on a real telescope.

Acknowledgements

For their support and encouragement we would like to thank Bill Borucki, Louis Boyd, David Genet, Russell Genet, Greg Henry, Butler Hine, Cindy Mason, George McCook, and Ann Patterson-Hine. For comments on a previous versions of this paper, we would also like to thank Bill Borucki, Russ Genet, and David Thompson. Lastly, thanks to Boris Rabin for assistance with the artwork.

References

[1] AutoScope. 1991. Robotic Observatory Control System Version 1.40 Software Reference Manual. AutoScope Corporation, Mesa, AZ.

[2] Boyd, L., D. Epand, D., Bresina, J., Drummond, M., Swanson, K., Crawford,D. Genet, D., Genet, R., Henry, G., McCook, G., Neely, W., Schmidtke, P., Smith, D., and Trublood, M. 1993. Automatic Instruction Set 1993. *International Amateur Professional Photoelectric Photometry (IAPPP) Communications*, No. 52, T. Oswalt (ed).

[3] Drummond, M. 1989. Situated Control Rules. *Proc. of Conf. on Principles of Knowledge Representation & Reasoning.* Toronto, Canada.

[4] Drummond, M., Swanson, K., and Bresina, J. (*forthcoming*). Robust Scheduling and Execution for Automatic Telescopes. In *Heuristic Scheduling Systems*, M. Fox & M. Zweben (eds). Morgan-Kaufmann.

[5] Genet, D. 1992. ATIScope software description document. AutoScope Technical Paper.

[6] Genet, R.M., and Hayes, D.S. 1989. *Robotic Observatories: A Handbook of Remote-Access*

APPENDIX: The ATIS93 Advice Statement

This appendix includes an modified excerpt from the current draft of the ATIS93 specification regarding the 116 statement. See Boyd, *et al.* (1993) for the full ATIS93 specification. This version of the 116 documentation includes an extension to allow the scheduler to insert planned "waits" into the schedule.

The 116 statement has been introduced in ATIS93 to allow several features not available in ATIS89. The primary new feature is to allow a programmatic approach to the selection of groups. The 116 statements should be treated as groups and may be transmitted during the night on systems which allow partial input files. 116 statements are usually generated by an APA rather than directly by a participating astronomer. Like other groups, the relative sequence within the ATIS input file is of no importance.

Identifier: 116
Information:

	Name	*Type*	*Input*	*Output*
1	Advice Number	integer	required	echoed
2	Start LST	real	required	echoed
3	End LST	real	required	echoed
4	Start UT	real	required	echoed
5	End UT	real	required	echoed
6	Previous Group	integer	required	echoed
7	Set Execution Count	integer	required	echoed
8	Group Number	integer	required	echoed
9	User Number	integer	required	echoed
10	Group Test	integer	required	echoed
11	Next If True	integer	required	echoed
12	Next If False	integer	required	echoed
13	Wait Flag	integer	required	echoed

Parameter Semantics:

Advice Number A unique (within a given telescope) positive integer assigned to each 116 statement.

Start LST The earliest Local Sidereal Time in decimal hours which yields true. Use 0.0 in both LST parameters for "don't care".

End LST The latest LST in decimal hours which yields true.

Start UT The earliest Universal Time in decimal hours which yields true. Use 0.0 in both UT parameters for "don't care".

End UT The latest UT in decimal hours which yields true.

Previous Group Constraint on the status of the previously executed group. The semantics of the parameter values are as follows: 0 indicates no constraint (*i.e.*, do not care whether or not the last group executed successfully); 1 indicates the last group must have executed successfully; and 2 indicates the last group must have aborted.

Set Execution Count Determines how to change the group execution count prior to executing the group. The execution count is not changed if a parameter value of -1 is given. A non-negative integer value will cause the execution count to be set to that value.

Group Number The 103 group to execute if tests pass. Selecting a non-existent group causes the 116 statement sequence to be exited, along with an error comment. If the group is executed, the execution count will be decremented whether it is successful or not.

User Number The 103 user number to check for a match. Group Number and User Number together always form a universally unique group to execute.

Group Test A boolean flag which indicates whether the 103 JD and Moon information should also be tested prior to executing the group. If the parameter value is 0, the 103 header tests are ignored. If the value is 1, then the JD and Moon group tests must also be true for the group to be enabled; failure of the group tests will cause the 116 statement to fail.

Next If True The Advice Number to execute if all tests pass and the group has been executed (successfully or not). Requesting Advice number zero to be used next causes the 116 statement sequence to be exited.

Next If False The Advice Number to execute if the tests fail. Requesting Advice number zero to be used next causes the 116 statement sequence to be exited.

Wait Flag A boolean flag (0 for False, 1 for True) that determines whether or not to wait until the current time is equal or greater than the Start LST and the Start UT. If the parameter is 1, then the wait is performed before the rest of the 116 applicability tests are processed.